Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

«Работа с IPython и Jupyter Notebook»

ОТЧЕТ
по лабораторной работе №1
дисциплины
«Технологии распознавания образов»

Выполнил:
Сотников Андрей Александрович
2 курс, группа ПИЖ-б-о-21-1,
011.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

_____
(подпись)

Проверил:

_____
(подпись)

Отчет защищен с оценкой_____Дата защиты_____

Ставрополь, 2022 г.

Цель работы: исследовать базовые возможности интерактивных оболочек IPython и Jupyter Notebook для языка программирования Python.

Проработка примеров из лабораторной работы:

Первая лабораторная работа (примеры из лабораторной работы)

In [1]:
```
3 + 2
```
Out[1]: 5

In [2]:
```
a = 5
b = 7
print(a + b)
```
```
12
```

In [4]:
```
n = 7
for i in range(n):
    print(i*10)
```
```
0
10
20
30
40
50
60
```

Рисунок 1 – Проработка примеров

In [5]:
```
i = 0
while True:
    i += 1
    if i > 5:
        break
    print("Test while")
```
```
Test while
Test while
Test while
Test while
Test while
```

In [6]:
```
from matplotlib import pylab as plt
%matplotlib inline
```

In [7]:
```
x = [i for i in range(50)]
y = [i**2 for i in range(50)]
plt.plot(x,y)
```
Out[7]: [<matplotlib.lines.Line2D at 0x23502438310>]

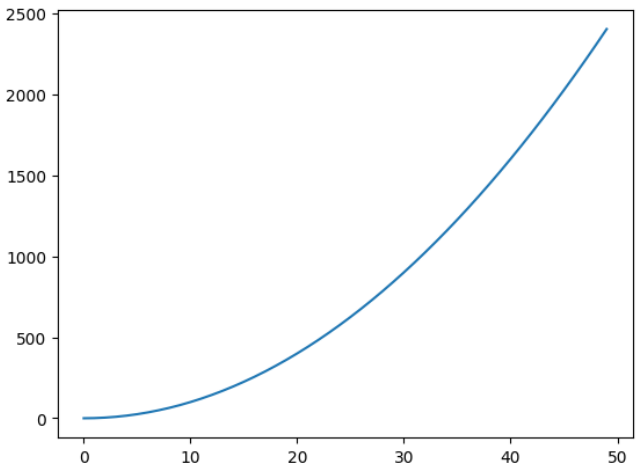


Рисунок 2 – Проработка примеров

In [9]:
```
%lsmagic
```

Out[9]:
```
Available line magics:
%alias  %alias_magic  %autoawait  %autocall  %automagic  %autosave  %bookmark  %cd  %clear  %cls  %colors  %conda  %config  %co
nnect_info  %copy  %ddir  %debug  %dhist  %dirs  %doctest_mode  %echo  %ed  %edit  %env  %gui  %hist  %history  %killbgscripts
%ldir  %less  %load  %load_ext  %loadpy  %logoff  %logon  %logstart  %logstate  %logstop  %ls  %lsmagic  %macro  %magic  %matpl
otlib  %mkdir  %more  %notebook  %page  %pastebin  %pdb  %pdef  %pdoc  %pfile  %pinfo  %pinfo2  %pip  %popd  %pprint  %precisio
n  %prun  %psearch  %psource  %pushd  %pwd  %pycat  %pylab  %qtconsole  %quickref  %recall  %rehashx  %reload_ext  %ren  %rep
%rerun  %reset  %reset_selective  %rmdir  %run  %save  %sc  %set_env  %store  %sx  %system  %tb  %time  %timeit  %unalias  %unl
oad_ext  %who  %who_ls  %whos  %xdel  %xmode

Available cell magics:
%%!  %%HTML  %%SVG  %%bash  %%capture  %%cmd  %%debug  %%file  %%html  %%javascript  %%js  %%latex  %%markdown  %%perl  %%prun
%%pypy  %%python  %%python2  %%python3  %%ruby  %%script  %%sh  %%svg  %%sx  %%system  %%time  %%timeit  %%writefile

Automagic is ON, % prefix IS NOT needed for line magics.
```

In [10]:
```
%%time
import time
for i in range(50):
    time.sleep(0.1)
```

```
CPU times: total: 31.2 ms
Wall time: 5.14 s
```

In [11]:
```
%timeit x = [(i**10) for i in range(10)]
```

```
1.45 µs ± 268 ns per loop (mean ± std. dev. of 7 runs, 1,000,000 loops each)
```

Рисунок 3 – Проработка примеров

Задания из прикреплённого файла lab_3.1:

Счастливый билетик

Билет считается счастливым, если выполнено следующее условие: сумма первых трёх цифр номера равна сумме последних трёх цифр.

Задание:

1) Определите число `ticket_number` — шестизначный номер билета;

2) Напишите код, который по шестизначному номеру `ticket_number` билетика проверяет, является ли он счастливым;

3) Если номер счастливый, выведите строку `Yes`, иначе — `No`.

Пример 1:

Input: 123456

Output: No

Пример 2:

Input: 123042

Output: Yes

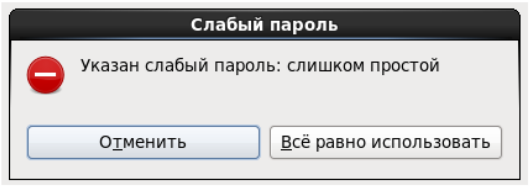
In [2]:
```
ticket_number = 123042
```

In [3]:
```
tick = list(map(int, str(ticket_number)))
print(tick)
if sum(tick[0:2+1]) == sum(tick[3:5+1]):
    print("Yes")
else:
    print("No")
```

```
[1, 2, 3, 0, 4, 2]
Yes
```

Рисунок 4 – Условие и решение задания №1

Пароль



Пусть пароль может содержать только латинские буквы, знаки препинания и цифры.

Пароль считается надёжным, если удовлетворяет следующим условиям:

- содержит буквы в разных регистрах;
- содержит цифры;
- содержит не менее `4` уникальных символов;
- не содержит ваше имя латинницей, записанное буквами любых регистров (anna, iVan, ...).

Иначе пароль считается слабым.

Задание:

1) Определите строку `password` — придуманный вами пароль;

2) Напишите код, который по паролю `password` проверяет, является ли он надёжным;

3) Если пароль надёжный, выведите строку `strong`, иначе — `weak`.

Пусть имя пользоватея -- Андрей.

Пример 1:

Input: Aandrei123

Output: weak

Пример 2:

Input: an12dRei

Output: strong

```
name = "Andrei"
password = "an12dRei"
```

```
if not(password == password.lower() or password.upper() == password or password.isalpha() or password.isdigit()
   or len(set(password)) < 4 or name.lower() in password.lower()):
  print("strong")
else:
  print("weak")
```

```
strong
```

Рисунок 5 – Условие и решение задания №2

Числа Фибоначчи

Как известно, числа Фибоначчи — это последовательность чисел, каждое из которых равно сумме двух предыдущих (первые два числа равны 1):

$$1, 1, 2, 3, 5, 8, 13, \ldots$$

Задание:

1) Определите число `amount` — количество чисел Фибоначчи, которые надо вывести;

2) Напишите код, который выводит первые `amount` чисел Фибоначчи.

Пример 1:

Input: 3

Output: 1 1 2

Пример 2:

Input: 10

Output: 1 1 2 3 5 8 13 21 34 55

In [6]:
```
amount = int(input("Ввелите число: "))
```

```
Ввелите число: 10
```

In [7]:
```
fibonach1 = fibonach2 = 1
print(fibonach1, fibonach2, end=' ')
for i in range(2, amount):
    fibonach1, fibonach2 = fibonach2, fibonach1 + fibonach2
    print(fibonach2, end=' ')
```

```
1 1 2 3 5 8 13 21 34 55
```

Рисунок 6 – Условие и решение задания №3

Пусть таблица `bikes.csv` содержит данные по арендам велосипедов за 2 года:

- `datetime` : дата и время аренды
- `season` : время года
- `temp` : температура воздуха по Цельсию
- `windspeed` : скорость ветра
- `registered` : число аренд

Одно из направлений исследования могло бы заключаться в проверке зависимости суммарного числа аренд от температуры воздуха.

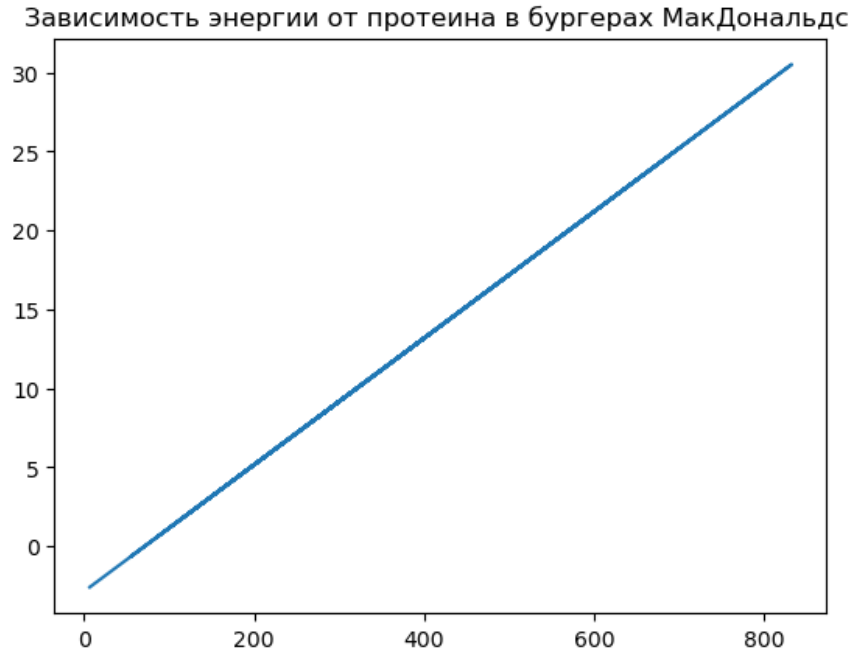
в такой ячейке (режим Markdown) можно писать текст

In [8]:
```
import csv
import statistics
from matplotlib import pylab as plt
%matplotlib inline

with open('mcd.csv', 'r', newline='', encoding='utf-8') as csvfile:
    reader = csv.reader(csvfile, delimiter=',')
    protein = []
    energy = []
    next(reader)
    for k in reader:
        protein.append(float(k[5]))
        energy.append(float(k[6]))

mid_protein = sum(protein)/len(protein)
mid_energy = sum(energy)/len(energy)

print(f"Среднее значение протеина: {mid_protein}")
print(f"Среднее значение энергии {mid_energy}")

statistic_protein = statistics.stdev(protein)
statistic_energy = statistics.stdev(energy)

print(f"Стандартное отклонение протеина: {statistic_protein}")
print(f"Стандартное отклонение энергии {statistic_energy}")

sum_protein = sum(protein)
sum_energy = sum(energy)

sm = 0
ssy = 0
for k, elem in enumerate(protein):
    sm += elem * energy[k]
    ssy += elem**2

a = len(protein)
b = (a * sm - sum_protein * sum_energy) / (a * ssy - sum_protein ** 2)
c = (sum_energy - b * sum_protein) / a

calc = []

for elem in protein:
    calc.append(b * elem + c)

print(f"Уравнение линейной зависимости: y = {b}x + {c}")
print(f"График функции методом наименьших квадратов: ")

plt.title("Зависимость энергии от протеина в бургерах МакДональдс ")
plt.plot(protein,calc)
```

Рисунок 7.1 – Решение задачи №4

```
Среднее значение протеина: 327.5038095238095
Среднее значение энергии 10.251666666666667
Стандартное отклонение протеина: 206.45313301443267
Стандартное отклонение энергии 10.042710963476694
Уравнение линейной зависимости: y = 0.040122048955289 27x + -2.888457212091351
График функции методом наименьших квадратов:
```

Out[8]: [<matplotlib.lines.Line2D at 0x20c71065c40>]



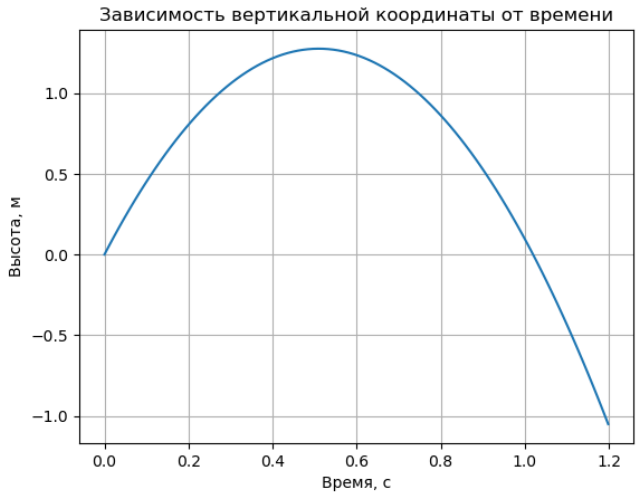
Чем больше в бургере из МакДональдс протеинов, тем больше больше в неём энергии.

Рисунок 7.2 – Решение задачи №4

Решение задачи, поставленной в методических указаниях:

Тело брошено со скоростью 10 м/с под углом 30° к горизонту. Через какое время оно будет на высоте 1,05 м?

In [6]:
```
import numpy as np
import matplotlib.pyplot as plt

v0 = 10
alpha = np.deg2rad(30)
h = 1.05
g = 9.81

v0_y = v0*np.sin(alpha)
t = (v0*np.sin(alpha) + np.sqrt((v0*np.sin(alpha))**2 + 2*g*h))/g

t_array = np.linspace(0, t, 100)
y_array = v0_y*t_array - (g*t_array**2)/2

plt.plot(t_array, y_array)
plt.xlabel("Время, с")
plt.ylabel("Высота, м")
plt.title("Зависимость вертикальной координаты от времени")
plt.grid()
plt.show()

print(f"Время полёта тела: {t:.2f} сек")
```




```
Время полёта тела: 1.20 сек
```

Рисунок 8 – Программа для решения выбранной задачи

Выводы: исследовали базовые возможности интерактивных оболочек IPython и Jupyter Notebook для языка программирования Python

Контрольные вопросы

1. Как осуществляется запуск Jupyter notebook?

При помощи команды jupyter-notebook в терминале IDE

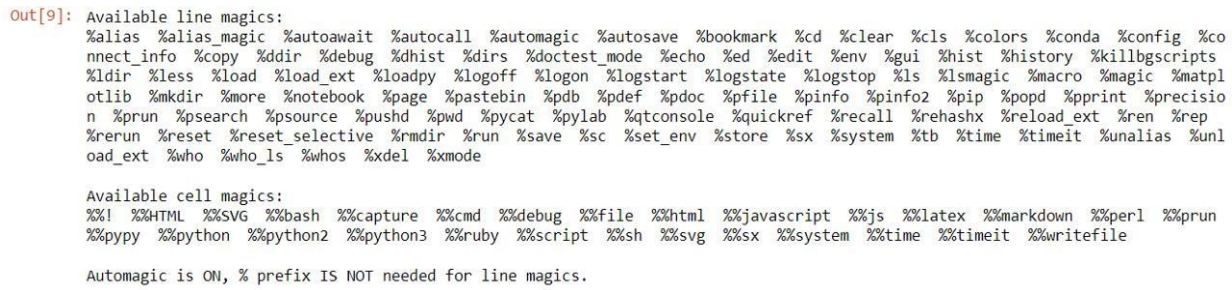
2. Какие существуют типы ячеек в Jupyter notebook?

Markdown и Code

3. Как осуществляется работа с ячейками в Jupyter notebook?

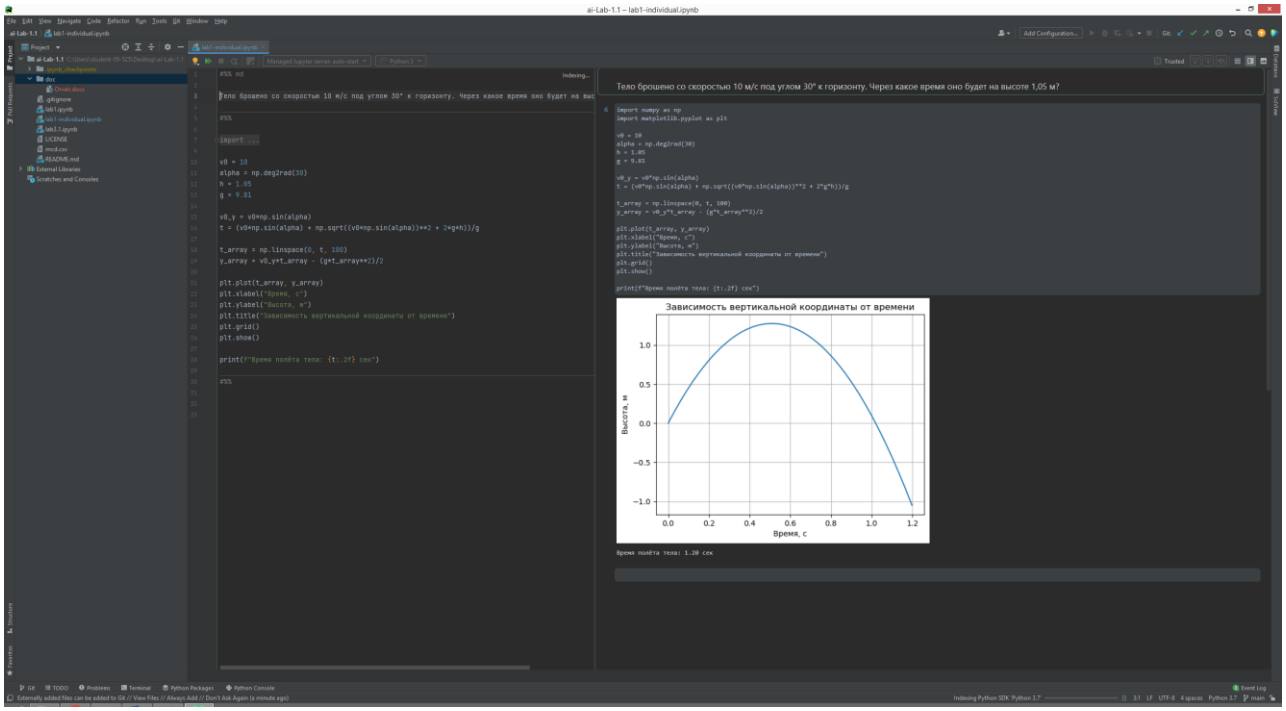
Ячейки в jupyter notebook можно создавать удалять и запускать их работу при помощи комбинаций клавиш

4. Что такое "магические" команды Jupyter notebook? Какие "магические" команды Вы знаете?



```
Out[9]: Available line magics:
%alias  %alias_magic  %autoawait  %autocall  %automagic  %autosave  %bookmark  %cd  %clear  %cls  %colors  %conda  %config  %co
nnect_info  %copy  %ddir  %debug  %dhist  %dirs  %doctest_mode  %echo  %ed  %edit  %env  %gui  %hist  %history  %killbgscripts
%ldir  %less  %load  %load_ext  %loadpy  %logoff  %logon  %logstart  %logstate  %logstop  %ls  %lsmagic  %macro  %magic  %matpl
otlib  %mkdir  %more  %notebook  %page  %pastebin  %pdb  %pdef  %pdoc  %pfile  %pinfo  %pinfo2  %pip  %popd  %pprint  %precisio
n  %prun  %psearch  %psource  %pushd  %pwd  %pycat  %pylab  %qtconsole  %quickref  %recall  %rehashx  %reload_ext  %ren  %rep
%rerun  %reset  %reset_selective  %rmdir  %run  %save  %sc  %set_env  %store  %sx  %system  %tb  %time  %timeit  %unalias  %unl
oad_ext  %who  %who_ls  %whos  %xdel  %xmode

Available cell magics:
%%!  %%HTML  %%SVG  %%bash  %%capture  %%cmd  %%debug  %%file  %%html  %%javascript  %%js  %%latex  %%markdown  %%perl  %%prun
%%pypy  %%python  %%python2  %%python3  %%ruby  %%script  %%sh  %%svg  %%sx  %%system  %%time  %%timeit  %%writefile

Automagic is ON, % prefix IS NOT needed for line magics.
```

5. Самостоятельно изучите работу с Jupyter notebook и IDE PyCharm и Visual Studio Code.



6. Приведите основные этапы работы с Jupyter notebook в IDE PyCharm и Visual Studio Code.

Pip install notebook, после чего можно запустить при помощи команды jupyter-notebook